VisualAge for Java: Un poderoso entorno visual de programación en Java.

Resumen

Desde el surgimiento del lenguaje de programación Java hacia inicios de la década de 1990, rápidamente se generaron grandes expectativas de programación alrededor de él y aunque Java no surgió como un lenguaje de propósito general, actualmente es utilizado en diversas y múltiples aplicaciones.

El contar con, y conocer un conveniente entorno de desarrollo, resulta de fundamental importancia para el rápido y adecuado desarrollo de aplicaciones. Así por ejemplo, está comprobado que en promedio, el código destinado a las GUI abarca aproximadamente alrededor del 55% del código total.

El presente trabajo presenta un panorama general de un poderoso entorno de programación, que agiliza el desarrollo de aplicaciones en Java.

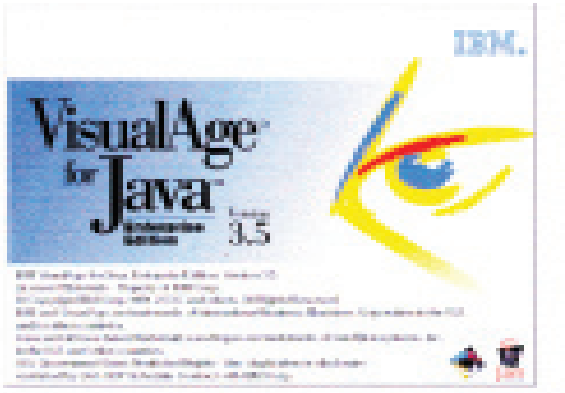
Palabras clave: Composición visual, GUI, IDE.

1.Introducción.

El presente trabajo no pretende ser un estudio profundo del entorno de programación VisualAge. Tiene mas bien la intención de presentar un panorama general del mismo, con la finalidad de considerarlo como una alternativa para aquellos que no lo conocen, así como el de presentar información derivada de la experiencia que se ha tenido al trabajar con él. Vale la pena mencionar también, que este trabajo tampoco es de ninguna manera un estudio comparativo, esto es, no se pretende decir que este entorno de programación en Java es mejor o peor que este otro, es solamente una descripción general de lo que se consideró como el funcionamiento y características más sobresalientes del entorno VisualAge.

El texto se desarrolla de la siguiente manera: primero se presenta un panorama general, en donde se describen las características más generales seguidas de una breve descripción de cada una de ellas, para después presentar algunos aspectos y peculiaridades que tiene, o que facilita VisualAge para el desarrollo de programas. La descripción de estos aspectos cubre un poco más a detalle, algunas de las características presentadas en el panorama general.

Es importante mencionar que la descripción que se hace en el presente trabajo, hace referencia al entorno de programación VisualAge for Java for Windows Enterprise Edition versión 3.5 (véase la Ilustración 1), ya que en el momento de escribir el mismo, se encontraba ya disponible la versión 4.0.



IAge for Java for Windows versión 3.5 Enterprise Edition.

Finalmente, se ha optado por utilizar el acrónimo VA en lugar de VisualAge for Java, debido a que esta última frase aparece repetidas veces en el texto.

2.Panorama general.

VA es un entorno integrado visual que soporta el ciclo completo de desarrollo de programas Java. Este entorno de programación proporciona, entre otras cosas, la facilidad para realizar las siguientes tareas:

- Rápido desarrollo de aplicaciones.
- Compilación incremental.
- Desarrollo basado en depósito de código (Repository).
- Creación de robustos programas en Java.

A continuación se describen brevemente cada uno de estos puntos.

Rápido desarrollo de aplicaciones.

Se pueden utilizar las características de programación visual de VA para desarrollar de manera rápida y sencilla tanto applets[1] como aplicaciones independientes. Ejemplo de ello es el editor de composición visual, ya que esta poderosa herramienta permite realizar, entre otras cosas:

- Diseñar la interfaz gráfica de usuario (GUI) para el programa.
- Especificar el comportamiento de los elementos de dicha interfaz.
- Definir y determinar la relación entre la interfaz de usuario y el resto del programa.

En base a lo anterior, VA genera el código necesario para implementar lo que se diseñó en el editor de composición visual, de hecho, en muchos casos, se pueden diseñar y ejecutar programas completos sin escribir una sola línea de código Java. Esto último es extremadamente ventajoso si se consideran las características, algunas de ellas intrincadas, que tiene la programación y manipulación directa de eventos y/o organizadores de diseño por ejemplo.

VA también proporciona asistentes[2] para la asesoría y la guía de diversas tareas, tales como la creación de nuevos applets, paquetes, clases, etcétera.

Compilación incremental

El entorno integrado de desarrollo (IDE) de VA compila automáticamente código fuente de Java y lo traduce en su correspondiente conjunto de bytecode. Cuando el código fuente (archivos .java) es importado hacia el espacio de trabajo (workspace) o añadido desde el depósito de código (repository), dicho código es compilado y analizado con respecto al contenido existente dentro del espacio de trabajo.

De esta manera, cuando se modifican, mueven, eliminan o copian elementos de programa, el código que es afectado es automáticamente recompilado para indicar los problemas que pudiesen ocurrir derivados de los cambios realizados.

Asimismo, si se introduce un error el IDE realiza la advertencia del mismo, además de proporcionar la opción para solucionarlo de manera inmediata o añadir el problema hacia la página de administración de errores para arreglarlo después.

Desarrollo basado en depósito de código (Repository).

Dentro de las características de particular interés con las que cuenta VA, se encuentra la de tener un sofisticado sistema de administración de código que facilita al desarrollador, la administración y mantenimiento de las múltiples ediciones o versiones de las clases, de los programas, o de los proyectos desarrollados.

Así por ejemplo, cuando se quiere "congelar" el estado del código en un punto dentro del proceso de desarrollo, ya sea por ser una edición funcional o por alguna otra razón, VA permite realizar una versión de la misma a través de un identificador, dicho identificador puede ser un nombre o un número proporcionado por el desarrollador o administrado por el sistema. Este proceso marcará a dicha edición como de sólo lectura, de tal manera que se pueden almacenar "instantáneas" de puntos importantes de verificación dentro del ciclo de desarrollo.

Creación de robustos programas en Java.

Con VA se pueden desarrollar programas bastante robustos, programas que, por llamarles de alguna manera, tienen un perfil industrial. Ejemplo de esto último son:

- Construir, modificar y usar JavaBeans[3].
- Revisar el código a diversos niveles de abstracción, esto es, a nivel de proyecto, de paquete, de clase o de método.

1 Un programa en Java diseñado para ser ejecutado dentro de un Web browser.
2 La literatura de VA tiende a llamarles SmartGuides, en este texto se opta por llamarles asistentes.
3 Es un modelo de componente reutilizable e independiente de plataforma.

- Usar el depurador visual integrado para examinar y actualizar el código mientras éste es ejecutado.
- Usar el depurador distribuido para depurar aplicaciones Java que son desarrolladas fuera del IDE.

La siguiente sección describe, algunas de las cualidades y particularidades que ofrece VA para el desarrollo de programas.

3.Desarrollo de programas

El entorno integrado de desarrollo (IDE) es un conjunto de ventanas que proporcionan el acceso hacia las herramientas de desarrollo de VA. Dentro de estas ventanas se encuentran:

1. Workbench: el workbench (refiérase a la Ilustración 2) constituye el área general de trabajo de VA, de tal manera que se puede considerar como la ventana principal de la aplicación, ya que es en ella en donde se pueden administrar las clases y los elementos de programa, así como abrir las demás ventanas como las del depurador, la consola, etcétera.

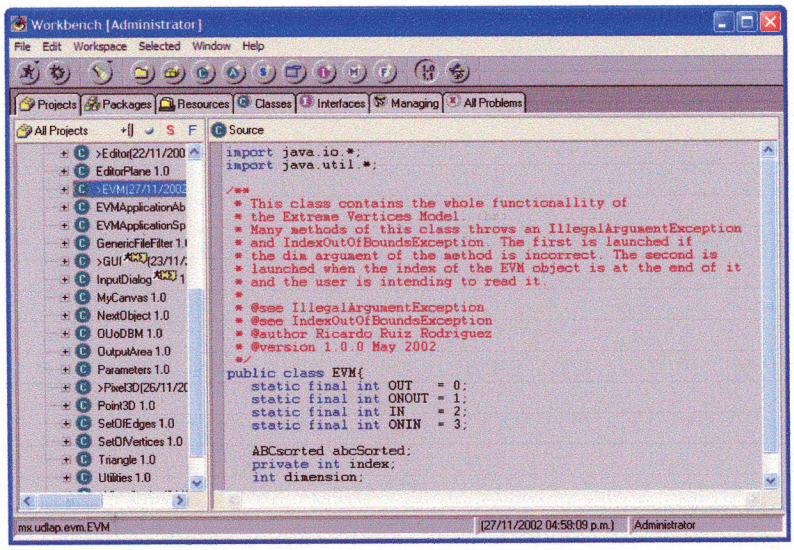


Ilustración 2. Área general de trabajo (Workbench) del entorno VisualAge for Java.

2. Repository Explorer: esta ventana administra y controla una de las características más importantes del desarrollo de programas Java dentro de VA, ya que funciona como un explorador del depósito de código, esto es, un explorador del conjunto de clases, métodos, etcétera, así como el conjunto de versiones que posean cada uno de estos elementos.
3. Log: la ventana de log es la que controla e indica entre otras cosas, los accesos que se han tenido al entorno de programación VA, así mismo, es en esta ventana en donde se indica si el depósito de código se encuentra o no en un estado consistente. La inconsistencia de depósito de código puede deberse por ejemplo, a una abrupta interrupción del suministro de energía de la máquina sobre la que se encuentra trabajando VA o a una inadecuada terminación de la sesión de trabajo.
4. Console: como su nombre lo indica, la consola es una ventana de salida de los datos que se procesan dentro de un código Java administrado por VA. Las instrucciones de salida de datos estándar de Java, mandarán su salida a la consola, así mismo, esta ventana captura las excepciones debidas a eventos no contemplados o no "cachados" durante la ejecución de una aplicación o un applet de Java.
5. Debugger: es la ventana que administra al depurador integrado de VA, el cual es una poderosa y útil herramienta que permite realizar la depuración de código Java de una manera sumamente eficiente. Más adelante, dentro de esta misma sección, se detalla un poco más las características principales del depurador.

El IDE puede ser usado para crear tanto aplicaciones independientes como applets de Java, y en general para desarrollar aplicaciones que involucren la escritura de código Java.

El editor de composición visual (Ilustración 3) puede ser usado para crear interfaces gráficas de usuario (GUI) a partir de beans previamente construidos o prefabricados, así como para definir las relaciones (denominadas como conexiones en VA) entre los diferentes beans que la componen. A este proceso de creación de programas orientados a objetos a través de la manipulación de representaciones gráficas de componentes se le denomina programación visual.

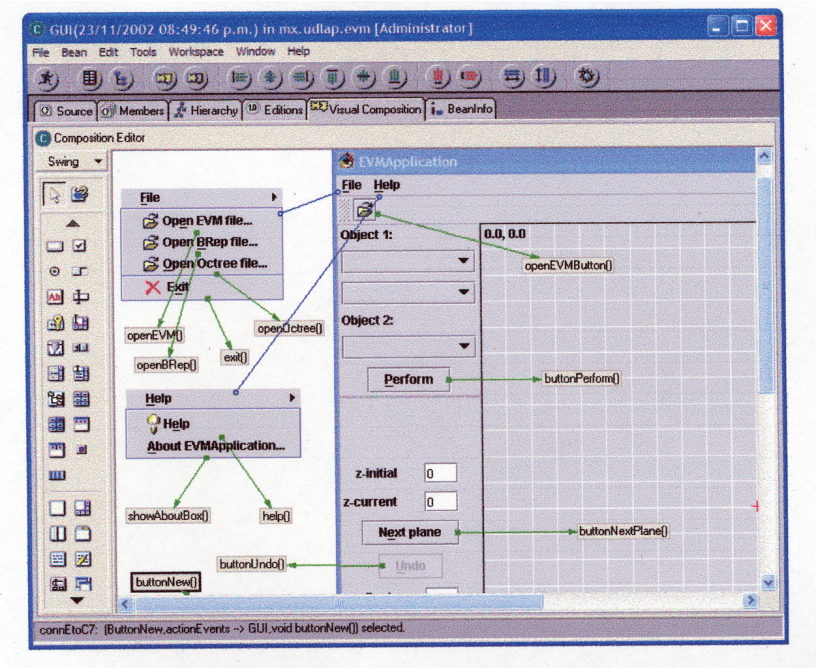
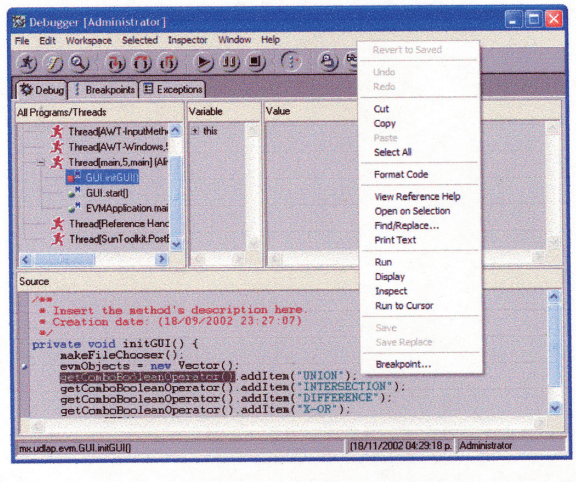


Ilustración 3. Pantalla del editor de composición visual de VisualAge utilizado para generar una GUI de una aplicación de modelado de sólidos.

Quienes conocen los aspectos de la programación de interfaces gráficas de usuario relacionados con los administradores de diseño de componentes, administradores de eventos y características Swing[4] por mencionar sólo algunas, saben de lo intrincado y laborioso que puede llegar a ser, pues bien, este entorno de programación permite desarrollar GUI completas sin escribir, en algunas ocasiones, una sola línea de código de ella. Esto último representa un considerable ahorro de tiempo de desarrollo en cuanto a GUI se refiere, y permite a los programadores centrarse en las tareas que se encuentran detrás de los datos y eventos que administran las GUI.

Ejemplo de ello por mencionar sólo uno, es un software de modelado de sólidos desarrollado en Java (Ruiz 2002). Para la construcción de las GUI de dicho software, una vez que se tenían los diseños en papel de lo que se quería, se consumieron aproximadamente 160 hrs. de desarrollo, utilizando solamente las características awt[5] de Java, mientras que en VA se construyó la misma GUI utilizando las características Swing de Java en aproximadamente 35 hrs. Obviamente este no es de ninguna manera un dato absoluto, ni tampoco significa que esa sea la proporción de ahorro de tiempo para todos los desarrolladores y para cualquier tipo de interfaz, es sólo un dato experimental en base a una experiencia particular de desarrollo que refleja un considerable ahorro de tiempo.

El depurador integrado de VA, permite entre otras cosas, la depuración de applets y aplicaciones que estén corriendo dentro del IDE. Dentro del depurador es posible por ejemplo, observar los hilos que se encuentran en ejecución, suspenderlos e inspeccionar los valores de sus variables, siempre y cuando dichos datos sean visibles.



La Ilustración 4 muestra una instantánea de la depuración de un método de inicialización de una de las clases de la GUI de la aplicación de modelado de sólidos mencionado con anterioridad. Paralelo al desarrollo del código que se esté realizando, es posible establecer puntos de ruptura para detener la ejecución en dicho punto, ya que todos los puntos de ruptura serán capturados por el depurador durante la ejecución del código. Así, durante el proceso de depuración es posible modificar el código, el cual será compilado automáticamente si así se desea, y ejecutado en la misma sesión de depuración. También es posible realizar evaluación de objetos y/o expresiones seleccionadas, observar y analizar los atributos (visibles) de un objeto determinado, etcétera, entre otras muchas características.

4 Swing es parte de la biblioteca de clases de Java, es una extensión del awt que ha sido integrada en Java 2 que ofrece una funcionalidad muy mejorada respecto del awt, ya que ofrece nuevos componentes, características ampliadas de los mismos, mejor manejo de eventos y una apariencia seleccionable.

5 Abstract Windows Toolkit.

Finalmente en cuanto al depurador se refiere, debe decirse que no es necesario eliminar los puntos de ruptura para la ejecución normal del programa, basta sólo con "apagar" la activación de los mismos para que todo funcione de manera normal y, que los puntos de ruptura establecidos con anterioridad, sigan siendo útiles en futuras sesiones de depuración.

Por otro lado, VA cuenta con un entorno de desarrollo de JSP/Servlet que permite el desarrollo, la ejecución y la prueba de archivos JSP[6] y Servlets[7]. Así mismo, el entorno de prueba WebSphere[8], el cual es instalado junto con el entorno de desarrollo de JSP/Servlet, proporciona soporte de servidor en tiempo de ejecución, para la prueba y depuración local de archivos JSP y Servlets. Los archivos JSP y los Servlets que se ejecuten de manera exitosa dentro de este entorno de prueba, también se ejecutarán sin problemas dentro del entorno de producción de servidor de aplicaciones WebSphere.

El asistente de creación de Servlets facilita el proceso de creación de Servlets. Así por ejemplo, cuando se desarrollan Servlets utilizando este asistente, se pueden usar archivos JSP que hereden de la clase PageListServlet, misma que proporciona características interesantes para algunos tipos de Servlets. Además de esto, se puede trabajar con los beans existentes y establecer conexiones entre ellos, entre otras múltiples opciones.

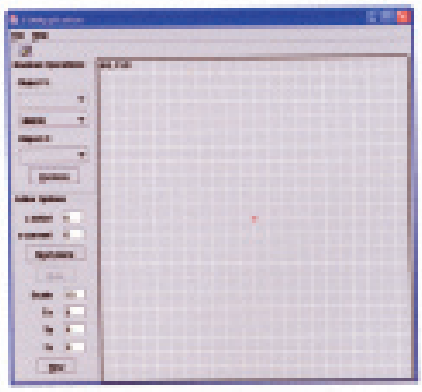


Ilustración 5. GUI principal de la aplicación de modelado de sólidos desarrollada con VisualAge

El programa de análisis (parser) de XML es un programa de análisis modular y de alto rendimiento escrito completamente en Java. Este programa proporciona la facilidad de brindarle a una aplicación, la cualidad de leer y escribir datos XML. Un archivo JAR[9] proporciona las clases necesarias para el análisis, la generación, la manipulación y la validación de documentos XML.

Estas no son todas las características con las que cuenta el entorno de desarrollo VA, son mas bien, algunas de las más representativas y útiles.

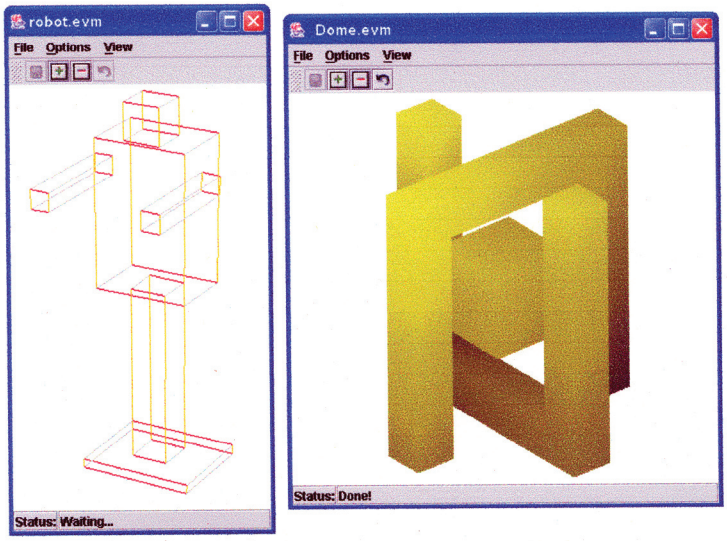


Ilustración 6. GUI de la ventana de visualización de la aplicación de modelado de sólidos desarrollada con VisualAge.

Casi para finalizar esta sección, en la Ilustración 5, la Ilustración 6 y la Ilustración 7, se presentan algunas de las interfaces principales, así como algunos de sus elementos de interacción para la aplicación de modelado de sólidos (Ruiz 2002) mencionada con anterioridad en esta misma sección. Esta aplicación fue desarrollada utilizando VA. Cabe mencionar, que algunas de las clases de esta aplicación que no tenían nada que ver con aspectos relacionados a la GUI, fueron desarrolladas con anterioridad al uso de, y de manera independiente a VA, y sin embargo, fueron reconocidas por este último sin ningún tipo de problema.

.

6 Java Server Pages es una tecnología de programación del lado del servidor, que permite incluir código Java dentro de páginas Web estáticas, y ejecutar dicho código cuando la página es solicitada.
7 Una clase del lado del servidor que responde a peticiones HTTP.
8 Un conjunto de productos de software de IBM, que ayuda a construir y administrar sitios Web de alto rendimiento.
9 Java ARchive es un formato de archivo independiente de plataforma que contiene muchos archivos integrados dentro de uno solo.

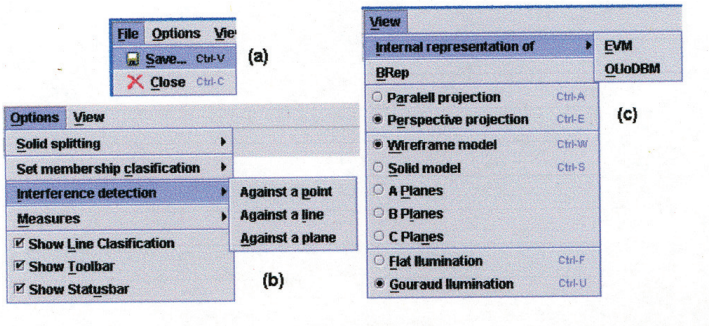


Ilustración 7. Algunos de los elementos de los menús de la GUI de la ventana de visualización: a) menú archivo, b) menú de opciones y c) menú de vista.

Por último, vale la pena mencionar que en la Universidad de Harvard, los estudiantes pueden registrarse en línea para sus cursos, usando una aplicación que fue construida con VisualAge for Java (IBM 2003).

En la siguiente sección se describen algunas de las características especiales con las que cuenta la versión Enterprise de VA.

4. Desarrollo de programas (versión Enterprise)

La versión Enterprise de VA tiene además de todo lo descrito en la sección de desarrollo de programas descrita con anterioridad, un grupo de herramientas y características importantes, de las cuales se hará una breve reseña de algunas de ellas en esta sección.

VA tiene un grupo de herramientas para XMI que proporciona un puente de integración entre VA y la herramienta de modelado Rational Rose. Esta característica permite generar código Java con las características de VA a partir de los modelos de Rational Rose, así como también generar modelos de Rational Rose derivados del código Java de VA, lo cual facilita y agiliza la transformación de los modelos de negocios.

El generador de XML es un programa en Java que se puede usar para editar un DDT[10], y generar documentos XML de muestra basados en dicho DDT. El crear este tipo de documentos puede ayudar a verificar y observar que el DDT esté trabajando de manera apropiada, así como a tener una mejor idea de que tipo de documentos XML son generados por el DDT.

Por otro lado, el entorno de desarrollo de los EJB[11] permite desarrollar beans que cumplan con las especificaciones de programación determinadas por Sun Microsystems. Este entorno de desarrollo para los EJB proporciona todo el soporte necesario en tiempo de ejecución para el servidor de aplicaciones WebSphere de IBM. Así mismo, un verificador incremental de consistencia, asegura que los EJB cumplan con las especificaciones de programación, además de indicar qué cambios se requieren para corregir las inconsistencias.

El constructor de persistencia por otro lado, permite establecer una correspondencia entre los objetos y las relaciones definidas entre ellos en información almacenada dentro de bases de datos relacionales, además proporciona los vínculos para la correspondencia existente entre los EJB y los datos relacionales.

Por último, dentro de las características que se mencionarán, el constructor de acceso de C++, proporciona acceso para los applets y/o aplicaciones Java a servicios escritos en el lenguaje de programación C++. Además, se pude usar para generar beans y envolturas (wrappers) C++ que permitan a los programas Java tener acceso a C++.

Vale la pena hacer nuevamente hincapié en que éstas no son todas las características con las que cuenta la versión Enterprise 3.5 de VA, sólo se han mencionado las que se han considerado por alguna u otra razón dentro de las más sobresalientes.

Finalmente, es de tomarse en cuenta que VA haya sido galardonado con distinciones tales como "Best Java IDE", "Best Team Development Tool", "Most Innovative Product", entre otros, por algunas de las principales revistas y grupos de desarrollo de software Java en los Estados Unidos de Norteamérica (IBM 2003).

5.Conclusiones

El surgimiento de Java en 1991, transformó de manera significativa el desarrollo de aplicaciones académicas y comerciales, ya que los lenguajes dominantes hasta ese momento eran el lenguaje C y C++. Con esto último, no se quiere decir que Java sea mejor que cualquier otro lenguaje de programación, sin embargo, es imposible negar el auge y popularidad creciente de Java desde su surgimiento.

10 Document Type Definition son las reglas que especifican la estructura de una clase particular de documentos SGML o XML.

11 Enterprise JavaBeans una arquitectura de componentes distribuidos definida por Sun Microsystems.

Java surge como un lenguaje concebido para correr en Web, aunque después "degeneró" para algunos, en un lenguaje de propósito general, de tal manera que actualmente resulta casi imposible enumerar las múltiples aplicaciones que se han construido basándose en este lenguaje de programación, debido principalmente quizá, a su relativa independencia de plataforma. La independencia de plataforma se debe al concepto de máquina virtual y, aunque éste no surge con Java, sí surge un conocimiento y una difusión más generalizada del mismo.

Tanto en el entorno académico como en el empresarial, la necesidad de realizar desarrollos tanto rápidos como de la mejor manera, resulta de fundamental importancia, y de ahí la necesidad de contar con un entorno de programación que proporcione, en la medida de lo posible, las herramientas y las facilidades para un rápido y mejor desarrollo.

VisualAge for Java no es de ninguna manera la panacea a todas las necesidades y complejidades subyacentes al proceso de desarrollo de software, pero sí se presenta como una alternativa eficiente, poderosa e interesante de desarrollo, para la mayoría de las tareas involucradas en el desarrollo de programas en Java.

6.Bibliografía

IBM
2000 IBM VisualAge for Java Online Help.

IBM
2003 VisualAge for Java product Overview and Developer Domain. http://www.3.ibm.com/software/awdtools/vajava.

Ruiz, R. R.
2002 Implementación del EVM (Extreme Vértices Model) en Java. Tesis de Maestría. Universidad de las Américas Puebla, Cholula, Pue.

Ricardo Ruiz Rodríguez